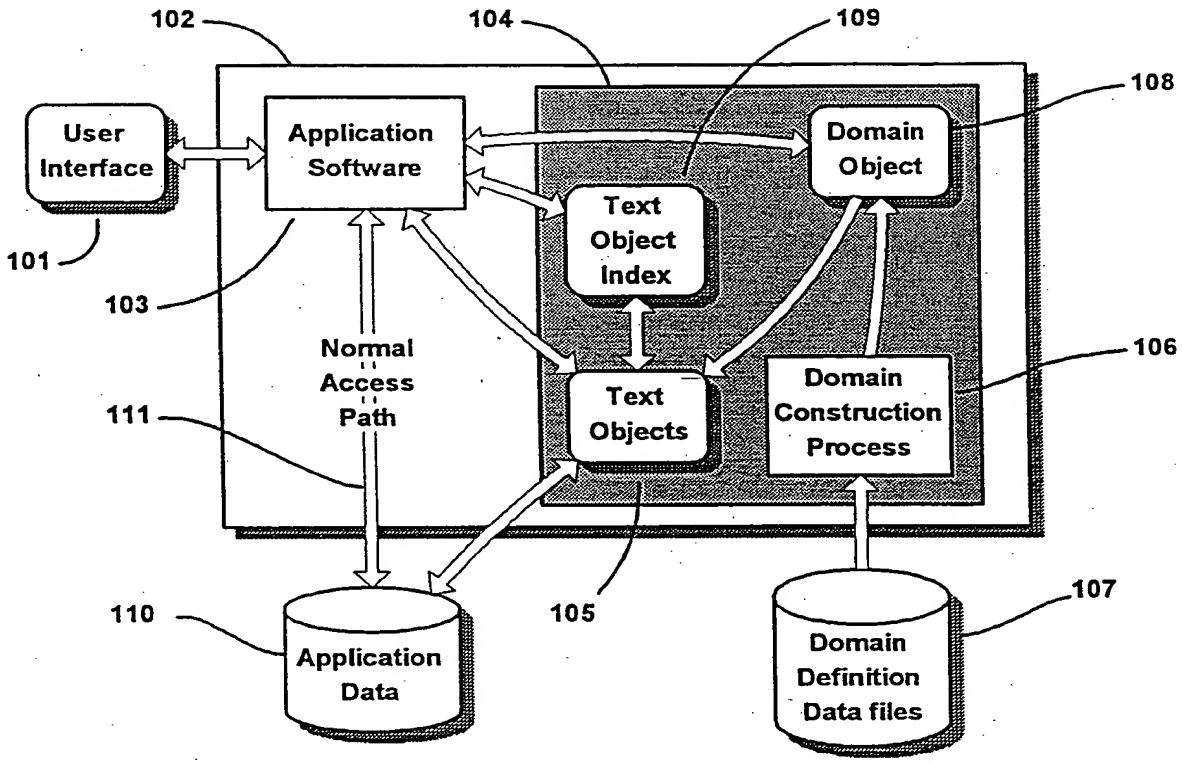
102
104
109
108
User Interface
Application Software
Domain Object
Text Object Index
101
103
Normal Access Path
106
Domain Construction Process
111
Text Objects
110
105
107
Application Data
Domain Definition Data files

Figure 1

201	76 Box Rd Townsville QLD
202	PO Box 92 Geelong VIC
203	39 Main St Box Hill VIC
204	8 Box Ave Devonport TAS
205	Cnr Box and Wolger Rds Townsville QLD
206	76 Box St Townsville NSW
207	231 Box Road Townsville QLD
208	53 3rd Ave, Townsville 4321 QLD
209	35 Third Avenue, Townsville Queensland 4321
210	333 Mt Pleasant Road, Springvale
211	191 Springvale Road, Mt Pleasant
212	123 Sydney Ave, Melbourne VIC

Figure 2

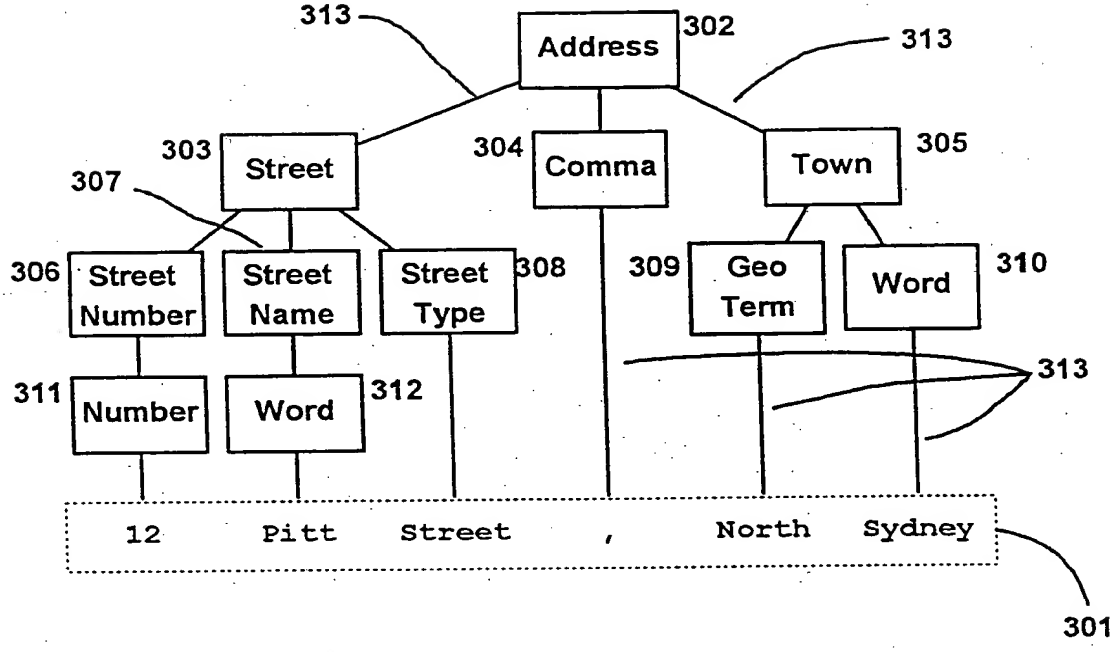
313
Address
302
313
303
Street
304
Comma
Town
305
307
306
Street Number
Street Name
Street Type
308
309
Geo Term
Word
310
313
311
Number
Word
312
12
Pitt
Street
,
North
Sydney
301

Figure 3

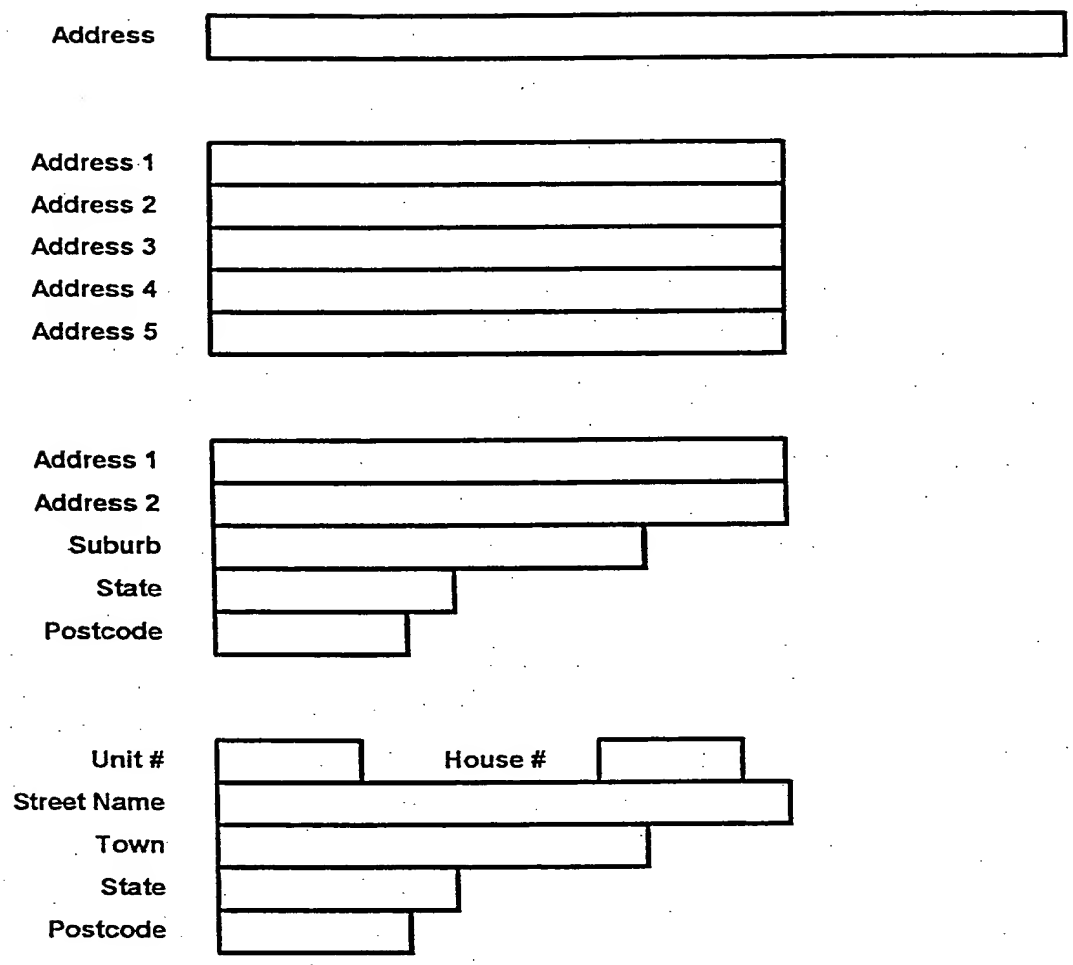
Address
Address 1
Address 2
Address 3
Address 4
Address 5
Address 1
Address 2
Suburb
State
Postcode
Unit #
House #
Street Name
Town
State
Postcode

Figure 4

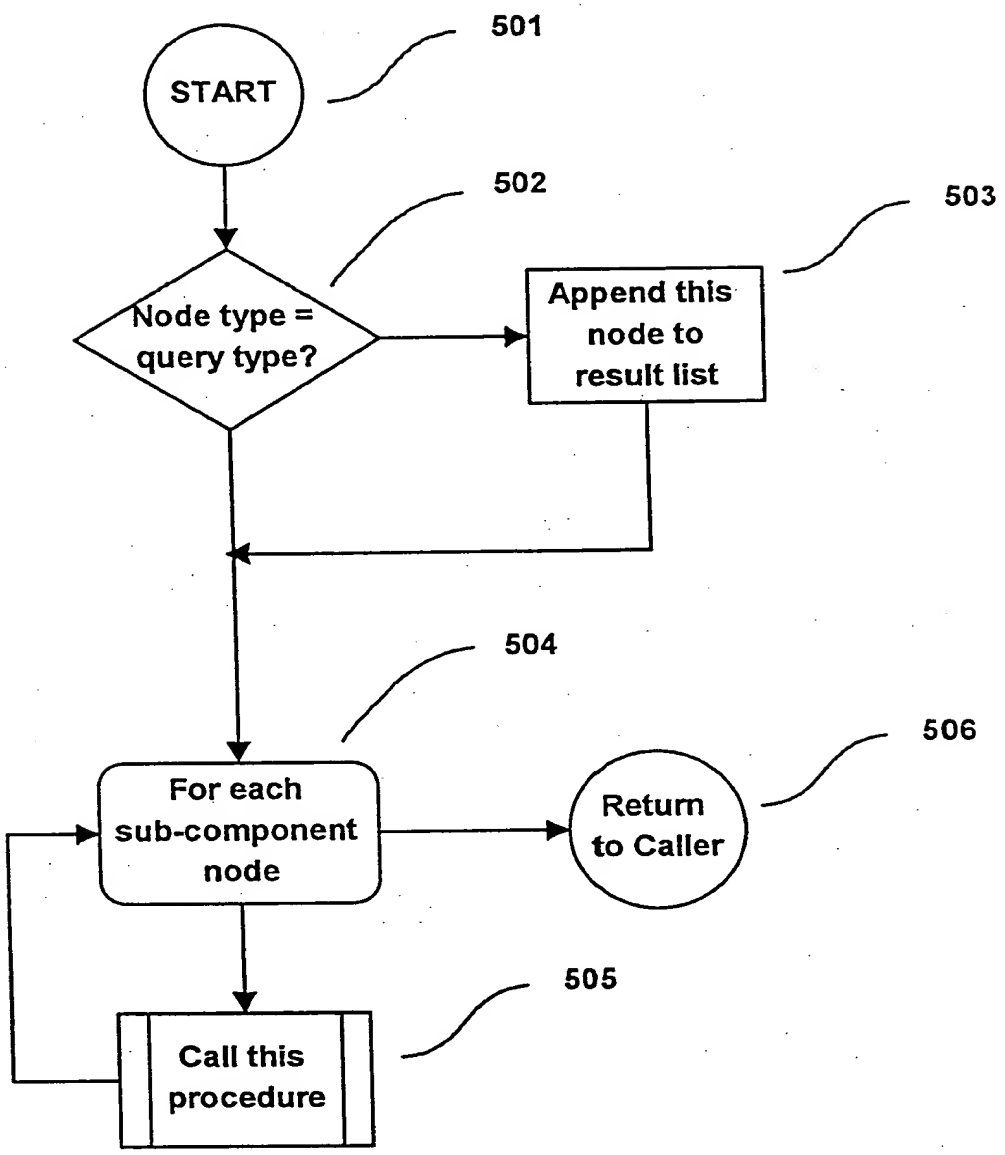
501
START
502
Node type = query type?
503
Append this node to result list
504
For each sub-component node
506
Return to Caller
505
Call this procedure

Figure 5

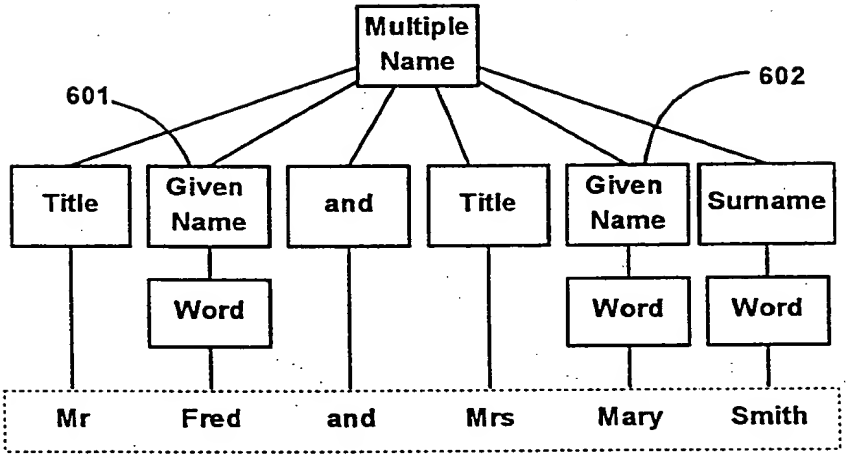
Multiple Name
601
602
Title
Given Name
and
Title
Given Name
Surname
Word
Word
Word
Mr
Fred
and
Mrs
Mary
Smith

Figure 6

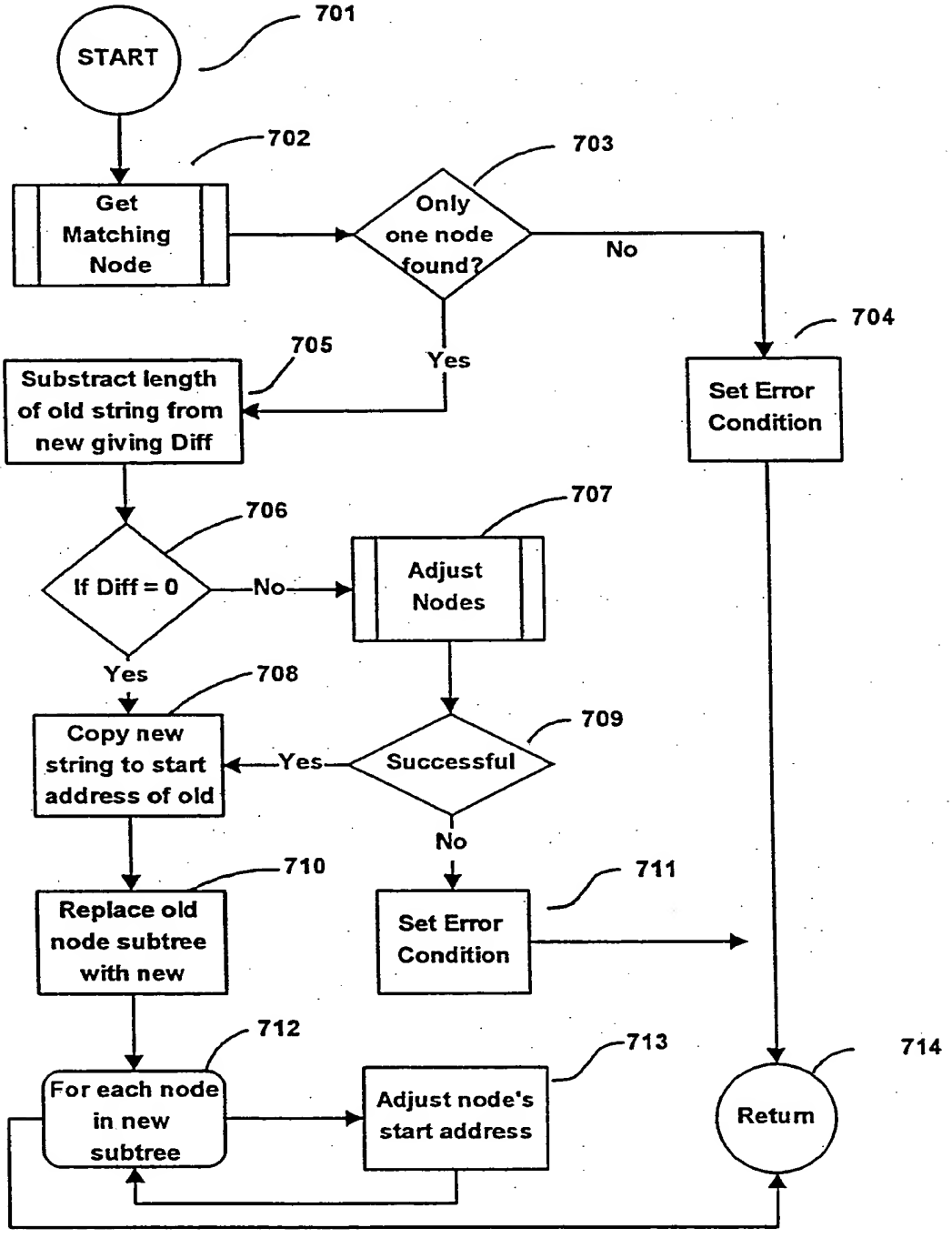
START
701
Get Matching Node
702
Only one node found?
703
No
Yes
Set Error Condition
704
Substract length of old string from new giving Diff
705
If Diff = 0
706
No
Adjust Nodes
707
Yes
Copy new string to start address of old
708
Yes
Successful
709
No
Replace old node subtree with new
710
Set Error Condition
711
For each node in new subtree
712
Adjust node's start address
713
Return
714

Figure 7

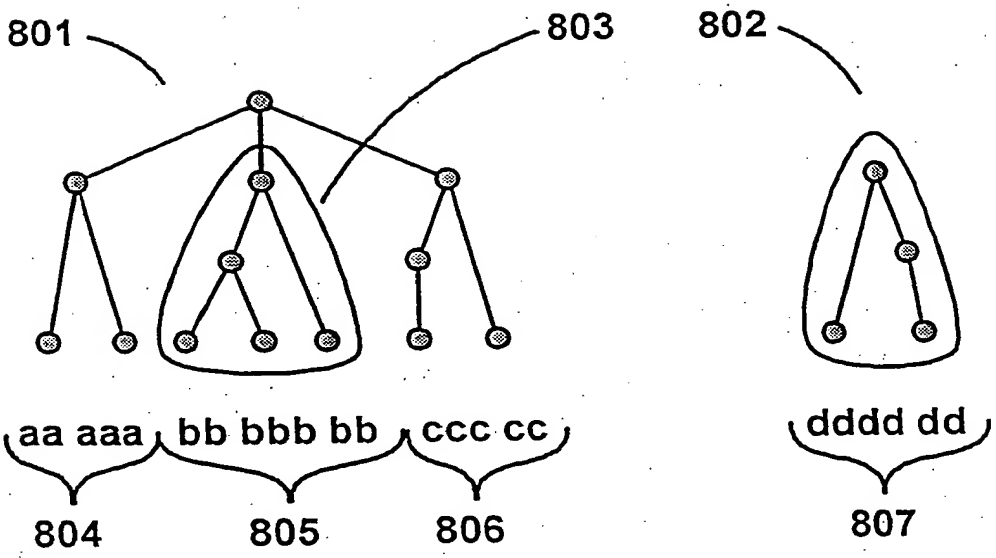

Figure 8

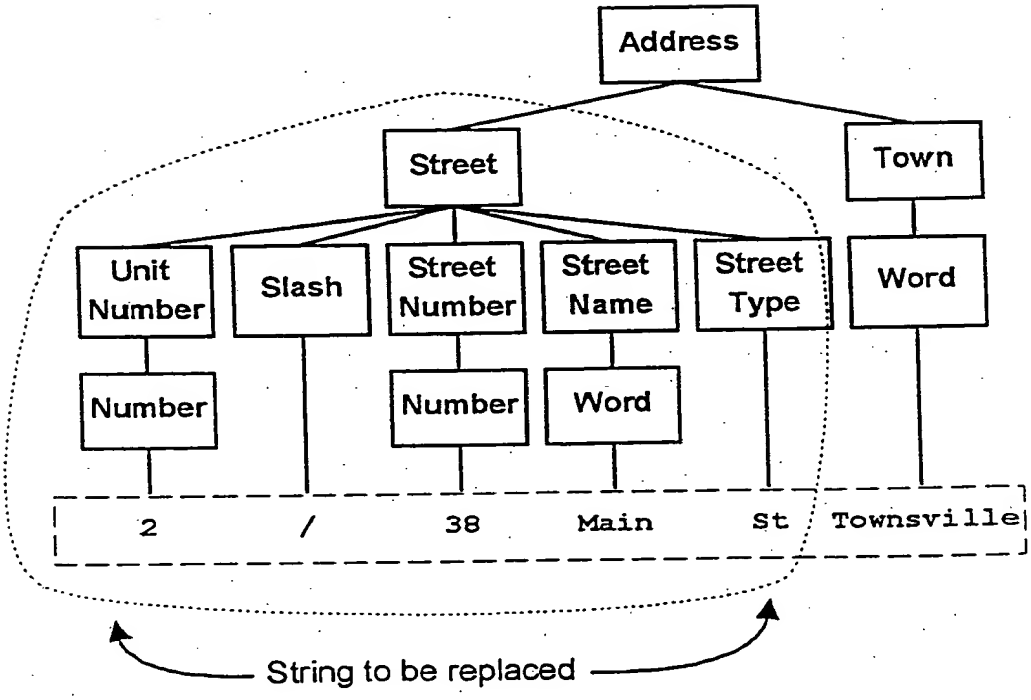

Figure 9

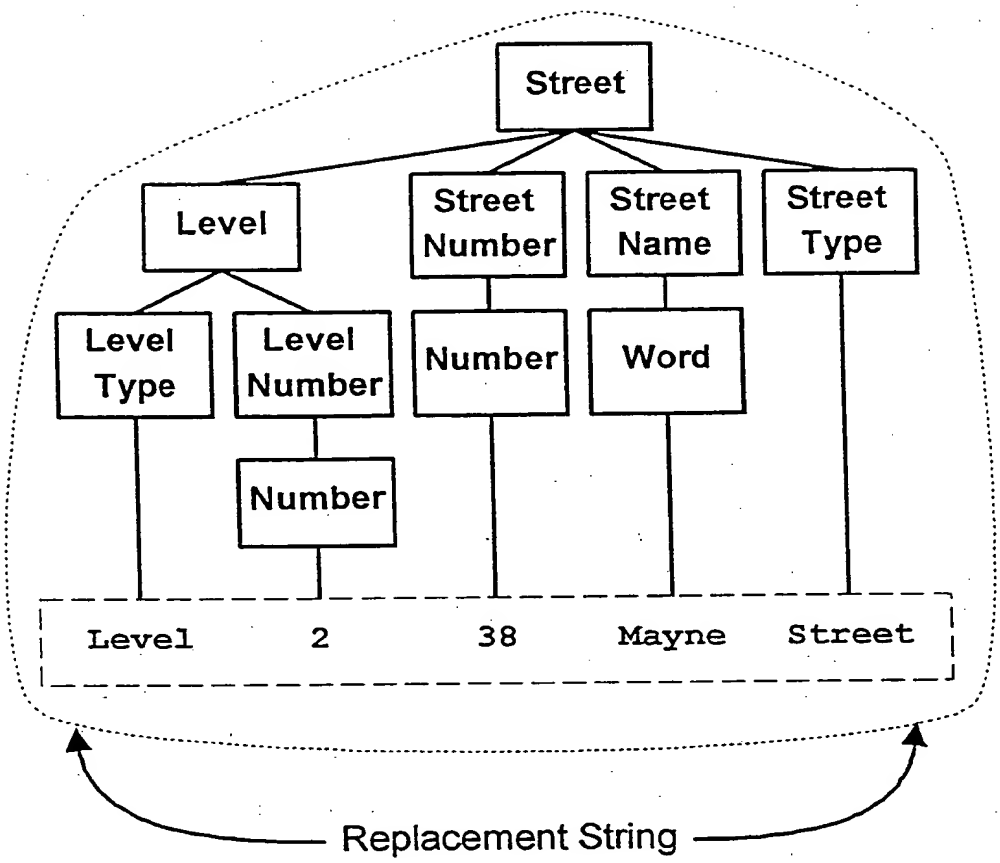
Street
Level
Street Number
Street Name
Street Type
Level Type
Level Number
Number
Word
Number
Level
2
38
Mayne
Street
Replacement String

Figure 10

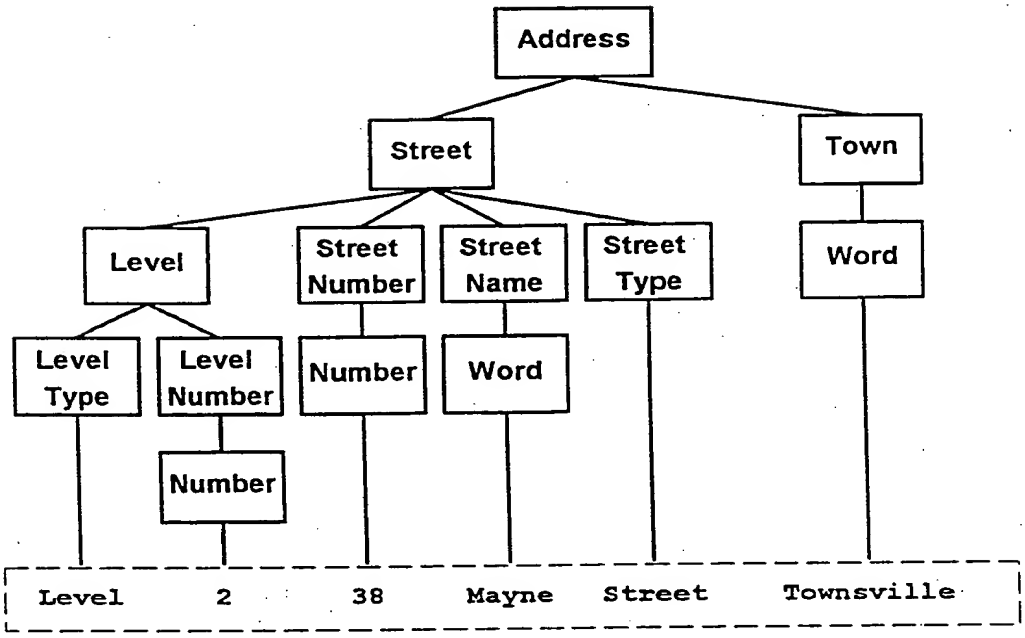
Address
Street
Town
Level
Street Number
Street Name
Street Type
Word
Level Type
Level Number
Number
Word
Number
Level
2
38
Mayne
Street
Townsville

Figure 11

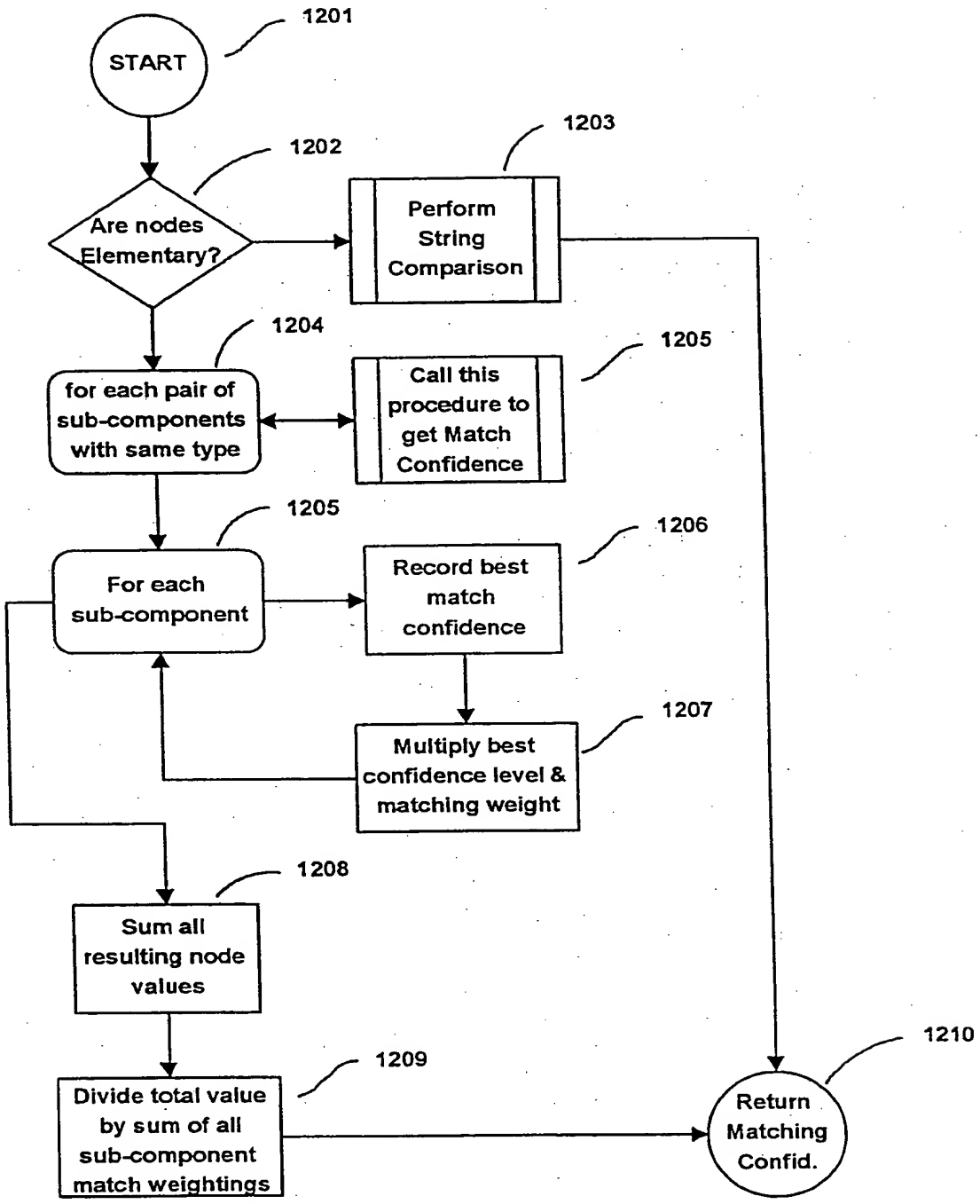
START
1201
Are nodes Elementary?
1202
Perform String Comparison
1203
for each pair of sub-components with same type
1204
Call this procedure to get Match Confidence
1205
For each sub-component
1205
Record best match confidence
1206
Multiply best confidence level & matching weight
1207
Sum all resulting node values
1208
Divide total value by sum of all sub-component match weightings
1209
Return Matching Confid.
1210

Figure 12

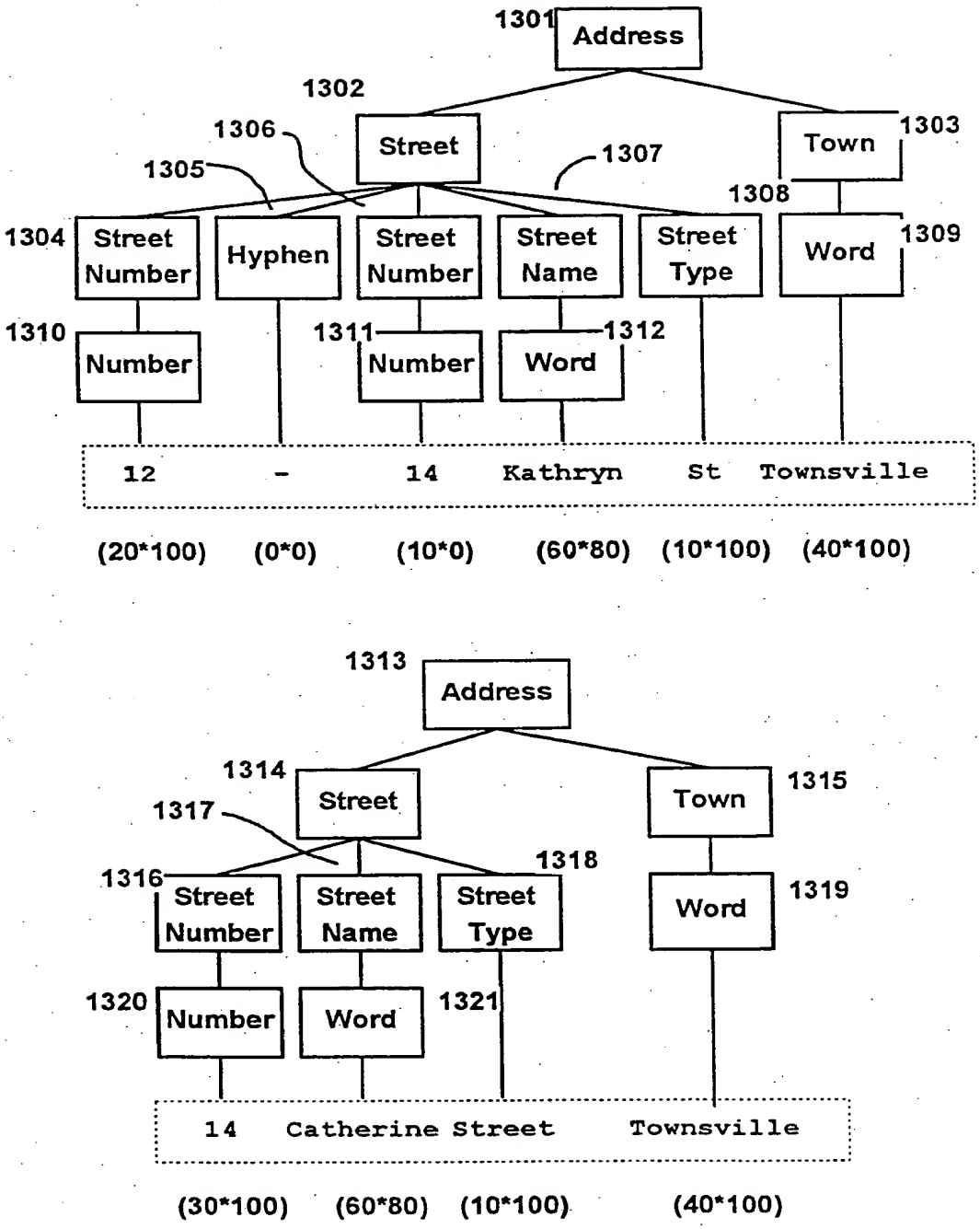
1301
Address
1302
Street
1303
Town
1306
1305
1307
1308
1304
Street Number
Hyphen
Street Number
Street Name
Street Type
Word
1309
1310
Number
1311
Number
Word
1312
12
-
14
Kathryn
St
Townsville
(20*100)
(0*0)
(10*0)
(60*80)
(10*100)
(40*100)
1313
Address
1314
Street
1315
Town
1317
1318
1316
Street Number
Street Name
Street Type
Word
1319
1320
Number
Word
1321
14
Catherine Street
Townsville
(30*100)
(60*80)
(10*100)
(40*100)

Figure 13

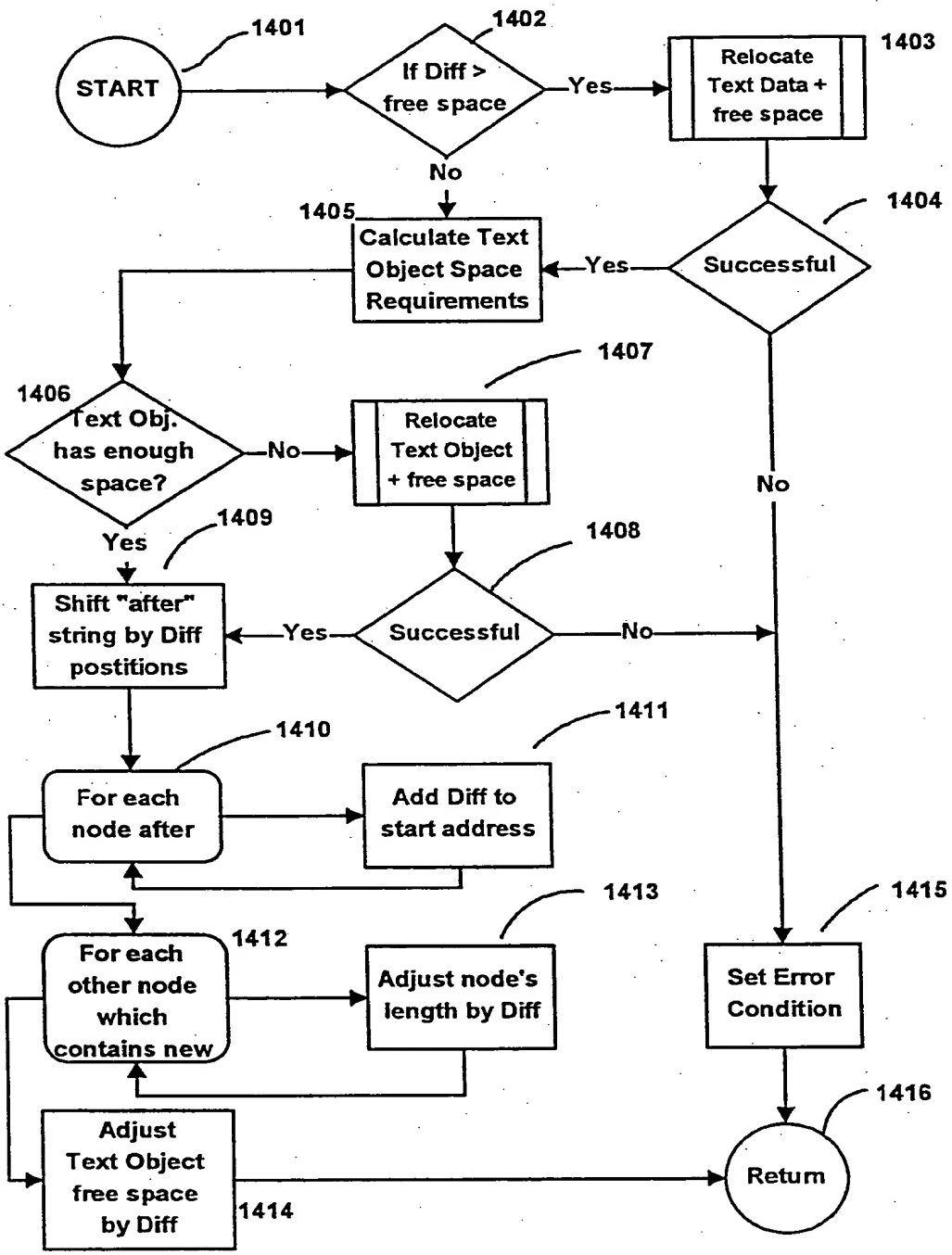
START
1401
If Diff > free space
1402
Yes
Relocate Text Data + free space
1403
No
1405
Calculate Text Object Space Requirements
Yes
Successful
1404
1406
Text Obj. has enough space?
No
Relocate Text Object + free space
1407
Yes
1409
Shift "after" string by Diff postitions
Yes
Successful
1408
No
No
1410
For each node after
Add Diff to start address
1411
1413
1412
For each other node which contains new
Adjust node's length by Diff
Set Error Condition
1415
1416
Adjust Text Object free space by Diff
1414
Return

Figure 14

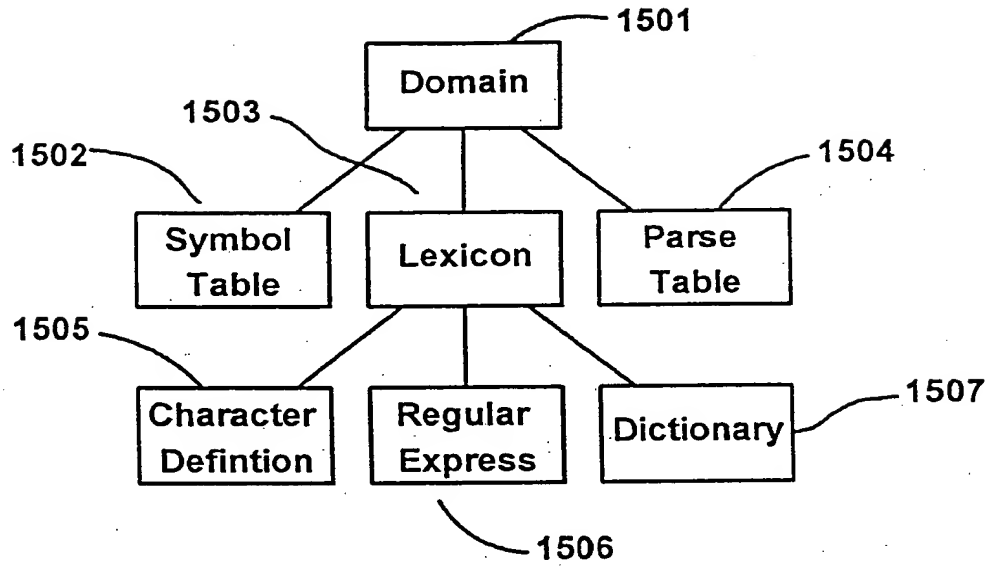

Figure 15

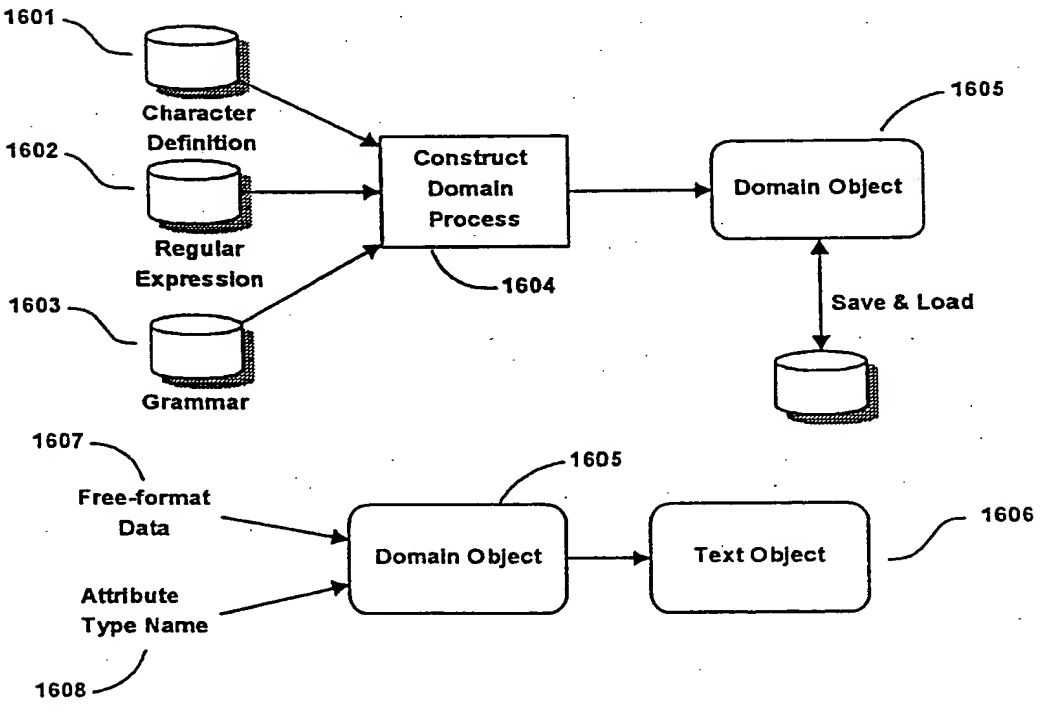

Figure 16

Standard Japanese Katakana Transliteration

a ア	i イ	u ウ	e エ	o オ
ka カ	ki キ	ku ク	ke ケ	ko コ
sa サ	si	su ス	se セ	so
ta	ti チ	tu	te	to ト
na ナ	ni	nu ヌ	ne	no ノ
ha	hi ヒ	hu フ	he ヘ	ho
ma マ	mi	mu ム	me メ	mo モ
ya ヤ		yu ユ		yo ヨ
ra	ri	ru	re レ	ro
wa ワ	wi		we ヱ	wo ヲ
ga	gi	gu	ge	go
za	zi	zu	ze	zo
da	di	du	de	do
ba	bi	bu	be	bo
pa	pi	pu	pe	po
n				

Standard Greek Transliteration

Α	α	a	Ι	ι	i	Ρ	ρ	r
Β	β	v	Κ	κ	k	Σ	σ	s
Γ	γ	g	Λ	λ	l	Τ	τ	t
Δ	δ	d	Μ	μ	m	Υ	υ	u
Ε	ε	e	Ν	ν	n	Φ	φ	f
Ζ	ζ	z	Ξ	ξ	x	Χ	χ	ch
Η	η	i	Ο	ο	o	Ψ	ψ	ps
Θ	θ	th	Π	π	p	Ω	ω	o

Figure 17

Sample Regular Expression Definition .

		Action **	alpha	digit	symbol	space	end of line	end of string
0	Error	0						
1	start	1	3	9	13	1	16	18
2	empty	4						
3	Initial	2	6	5	5	4	5	5
4	Initial space+	3	5	5	5	4	5	5
5	Initial ^space	5						
6	Word+	2	6	8	8	7	8	8
7	Word+ space	3	8	8	8	7	8	8
8	Word+ ^space	7						
9	0-9	2	12	10	12	11	12	12
10	0-9+	2	12	10	12	11	12	12
11	0-9+ space	3	12	12	12	11	12	12
12	0-9+ ^space	8						
13	sym	2	15	15	15	14	15	15
14	sym space	3	15	15	15	14	15	15
15	sym ^space	10						
16	eol+ space*	3	17	17	17	16	16	18
17	eol+ ^space	11						
18	eoi	9	0	0	0	0	0	0

** Action	Action Description
0	Error in Table
1	Bypass leading spaces
2	Append this character to character buffer
3	Append trailing space to character buffer
4	Empty string
5	Create "initial" token; go back 1 char; set state to 1
7	Create "word" token; go back 1 char; set state to 1
8	Create "number" token; go back 1 char; set state to 1
9	Create "end of input" token; go back 1 char; set state to 1
10	Create "symbol" token; go back 1 char; set state to 1
11	Create "end of line" token; go back 1 char; set state to 1

Figure 18

```
Address
  -> StreetAddr, Town Zipcode State
   | PostBox, Town Zipcode State ;
StreetAddr
  -> Street
   | StreetNum Street
   | AptType AptNum StreetNum Street
   | StreetNum Street AptType AptNum ;
Street
  -> StreetName StreetType StreetDir        :-2
   | StreetName StreetType ;
StreetName
  -> word | word word ;
StreetNum -> nbr ;
AptNum -> nbr ;
StreetType                                  =-4=
  -> "Ave"    | "Avenue"   ("Ave")
   | "Rd"     | "Road"     ("Rd")
   | "St"     | "Street"   ("St") ;
StreetDir
  -> Geo ;
Geo
  -> "North" | "N" ("North")
   | "South" | "S" ("South")
   | "East"  | "E" ("East")
   | "West"  | "W" ("West") ;
AptType                                     ==
  ->  "Apt"  | "Apartment"
   |  "Unit"
   |  "Suite" |  "Ste" ;
Zipcode
  -> 99999    | 99999 "-" 9999 ;
PostBox
  -> PostPref PostBoxNum ;
PostPref                                    =-9=
  -> "PO Box" | "Box" ;
PostBoxNum
  -> nbr | nbr A | A nbr ;
Town
  -> word | word word
   | Geo word | Geo word word ;
State
  -> "ALABAMA"      ("AL") | "AL"
   | "ALASKA"       ("AK") | "AK"
   | "ARIZONA"      ("AZ") | "AZ"
   | "ARKANSAS"     ("AR") | "AR"
   | "CALIFORNIA"   ("CA") | "CA"  ;
```

Special Symbols:

->	Consists of
|	Or
;	Rule terminator
("..")	Matching equivalence
=+9=	Matching Significance Adjustment
==	Zero Matching Significance
:+9	Parsing Significance Adjustment

Reserved Words:

word	one or more letters
nbr	one or more digits
A	one letter
9	one digit
,	comma or nothing

Figure 19

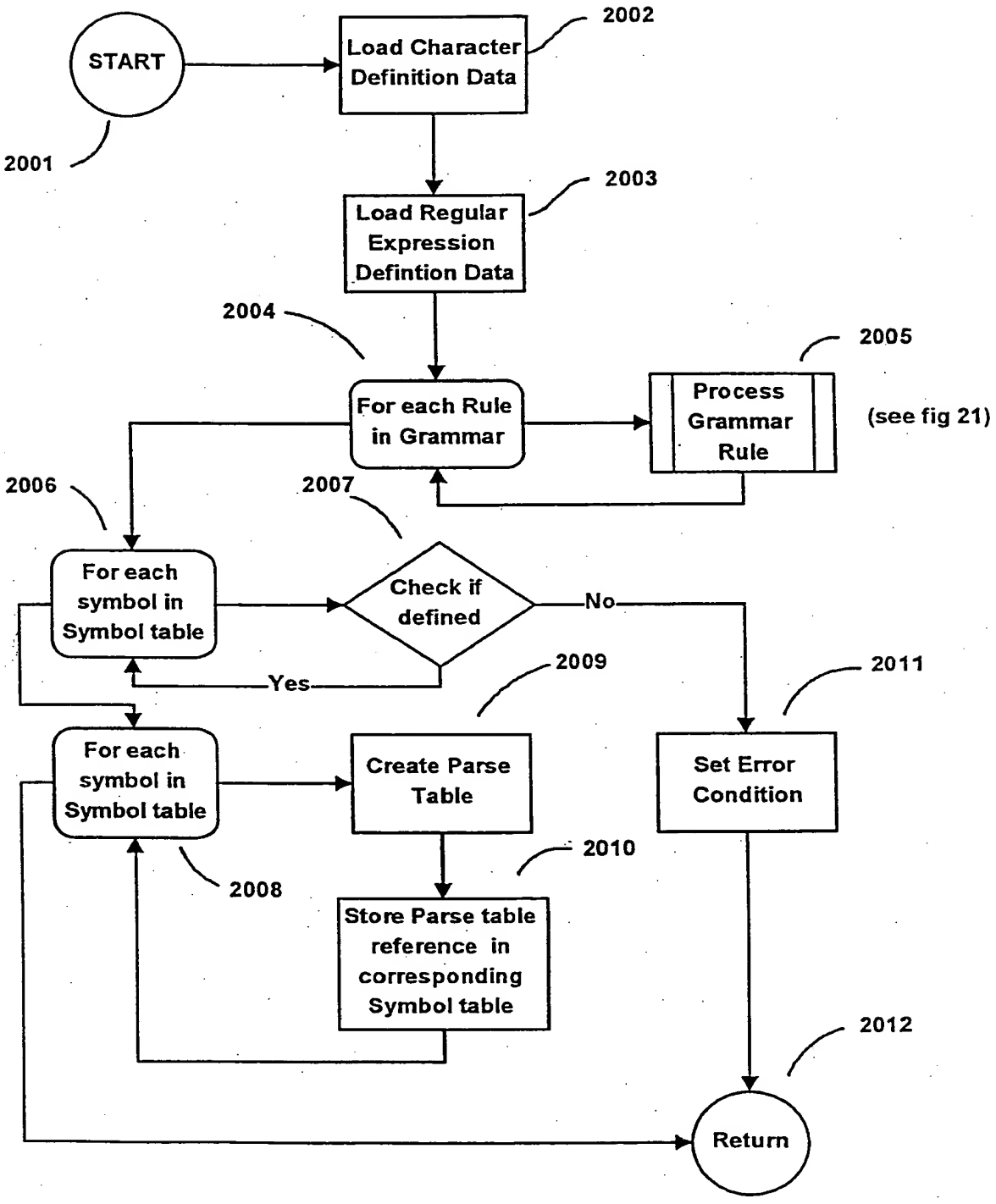
START
2001
Load Character Definition Data
2002
Load Regular Expression Defintion Data
2003
2004
For each Rule in Grammar
2005
Process Grammar Rule
(see fig 21)
2006
For each symbol in Symbol table
2007
Check if defined
No
Yes
2009
For each symbol in Symbol table
Create Parse Table
2011
Set Error Condition
2008
2010
Store Parse table reference in corresponding Symbol table
2012
Return

Figure 20

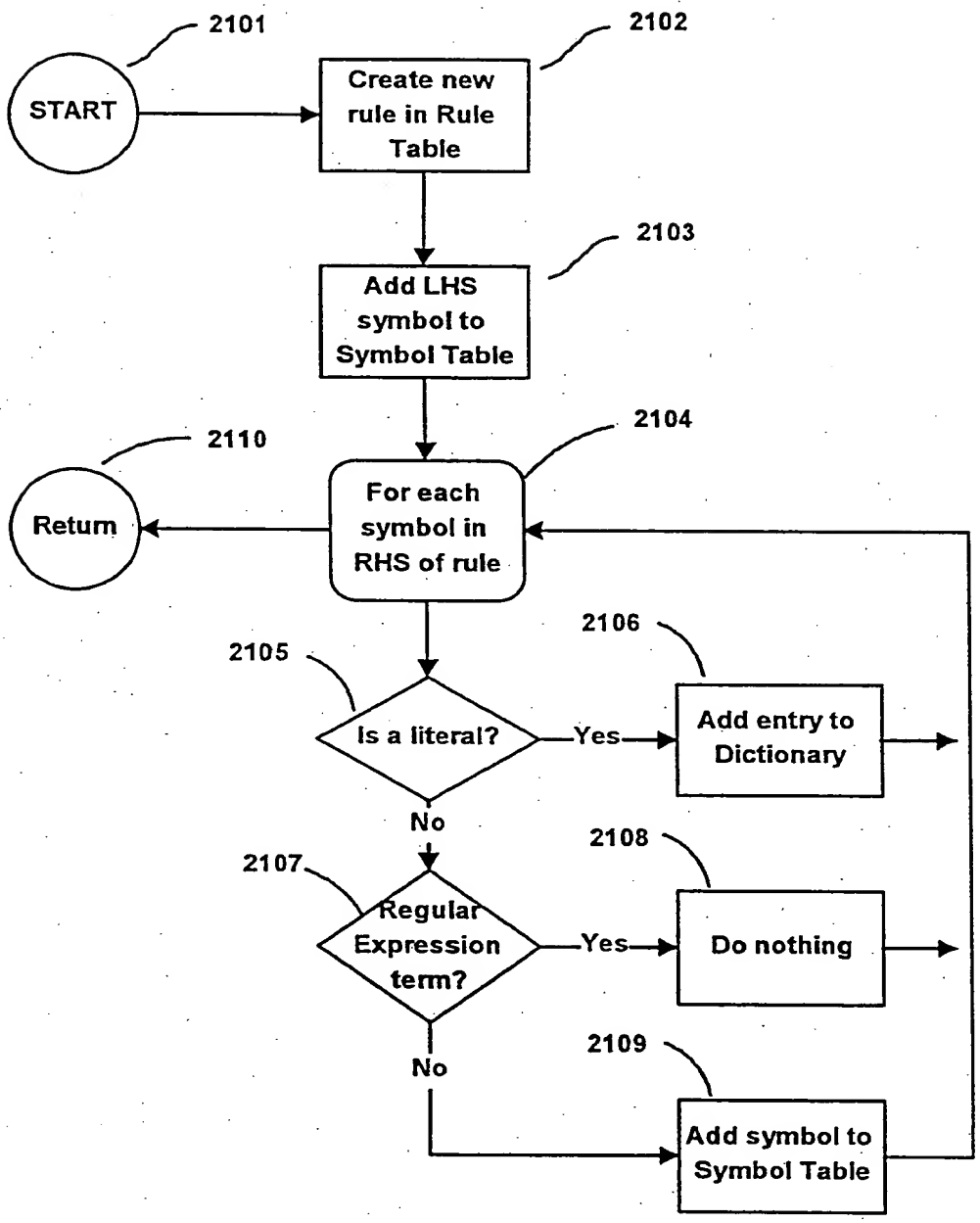
2101
START
2102
Create new rule in Rule Table
2103
Add LHS symbol to Symbol Table
2104
For each symbol in RHS of rule
2110
Return
2105
Is a literal?
Yes
2106
Add entry to Dictionary
No
2107
Regular Expression term?
Yes
2108
Do nothing
No
2109
Add symbol to Symbol Table

Figure 21

SQL Database Implementation Example

```
1.  CREATE DOMAINOBJECT US_ADDRESS ;
2.  UPDATE US_ADDRESS
      SET LANGUAGE = EXTERNAL 'Path/English.txt',
          GRAMMAR  = EXTERNAL 'Path/US_Addr.txt' ;

3.  CREATE TEXTOBJECT ADDRESS ;
4.  UPDATE US_ADDRESS
      SET DOMAIN = US_ADDRESS,
          TYPE = "Address" ;

5.  CREATE TABLE PERSONS(
      Name            CHAR (20),
      Home_Addr       ADDRESS ) ;

6.  INSERT INTO PERSONS ( Name, Home_Addr )
      VALUES (
        "John Smith",
        "123 Cathy Street, Apt 5, Huntsvale, CA, 98765" ) ;

7.  SELECT FROM PERSONS
      WHERE Home_Addr = "Unit 5 123 Cathy St, Huntsvale, CA" ;

8.  SELECT FROM PERSONS
      WHERE Home_Addr.State = "California" ;

9.  SELECT FROM PERSONS
      WHERE Home_Addr.Street MATCHES "Kathie St" > 0.80 ;
```

New Reserved Words:

DOMAINOBJECT, TEXTOBJECT, LANGUAGE,
GRAMMAR, TYPE, MATCHES

Figure 22